Watch Dog Timer (WDT)功能使用說明(使用內部 250KHz)

適用產品：SM59264、SM59128、SM8954A、SM8958A、SM89516A、SM894051、SM79108、SM79164、SM59D03/04G2

應用說明：

1 針對使用新茂（SyncMOS）MCU，如有因程式設計之關係，需防止當機之發生時，可啓動 Watch Dog 功能，以防止當機之發生。

2 此 WDT 功能共有八組重置時間可供選擇。(SM8954A/58A/516A 重置時間乘 8 倍)

3 此 WDT 功能爲計數器 0，計數器 1，計數器 2 外之另一組利用內部 RC 震盪的 16 位的計數器。

4 應用範圍：1、防當機裝置。2、自動 S/W Reset 功能。

使用方法：

1. 特殊功能暫存器說明：

1.1. WDT 控制暫存器(WDTC, $9F)

	bit-7							bit-0
	WDTE	Unused	Clear	Unused	Unused	PS2	PS1	PS0
Read / Write:	R/W	-	R/W	-	-	R/W	R/W	R/W
Reset value:	0	*	0	*	*	0	0	0

WDTE：WDT 致能位

當設爲 1 時則啓動 WDT 計時功能，初使值爲 0。

當 WDT Reset 發生時，此位會被清成 0，故如果要再次啓動 WDT，則需再一次將此位設成 1

Clear：WDT 重置位

當設爲 1 時則是將 WDT 16 位計數值清除爲 0，當計數值清除爲 0 後，CLEAR 會自動恢復爲 0。

型號: SM59264、SM59128、SM894051、SM79108、SM79164

PS【2：0】：WDT計數溢位週期選擇

PS【2:0】	**000**	**001**	**010**	**011**	**100**	**101**	**110**	**111**
溢位週期	**2.048ms**	**4.096ms**	**8.192ms**	**16.384ms**	**32.768ms**	**65.536ms**	**131.072ms**	**262.144ms**

型號: SM8954A、SM8958A、SM89516A

PS【2：0】：WDT 計數溢位週期選擇

PS【2:0】	**000**	**001**	**010**	**011**	**100**	**101**	**110**	**111**
溢位週期	**16.38ms**	**32.76ms**	**65.53ms**	**131.07ms**	**262.14ms**	**524.28ms**	**1048.57ms**	**2097.15ms**

WDT 控制暫存器(WDTC, $8E)(SM59D03/04G2 設定)

	bit-7							bit-0
	WDTE	Unused	Clear	Unused	Unused	WDTM2	WDTM1	WDTM0
Read / Write:	R/W	-	R/W	-	-	R/W	R/W	R/W
Reset value:	0	*	0	*	*	0	0	0

WDTE：WDT 致能位

當設爲 1 時則啓動 WDT 計時功能，初使值爲 0。

當 WDT Reset 發生時，此位會被清成 0，故如果要再次啓動 WDT，則需再一次將此位設成 1

Clear：WDT 重置位

當設爲 1 時則是將 WDT 16 位計數值清除爲 0，當計數值清除爲 0 後，CLEAR 會自動恢復爲 0。

型號: SM59D03/04G2

WDTM【2：0】：WDT計數溢位週期選擇

WDTM【2:0】	000	001	010	011	100	101	110	111
溢位週期	**1.58ms**	**3.15ms**	**6.30ms**	**12.60ms**	**25.12ms**	**50.41ms**	**100.82ms**	**201.65ms**

1.2. WDT Key 暫存器- (WDTKEY, $97)

	bit-7							bit-0
	WDT KEY7	WDT KEY6	WDT KEY5	WDT KEY4	WDT KEY3	WDT KEY2	WDT KEY1	WDT KEY0
Read / Write:	W	W	W	W	W	W	W	W
Reset value:	0	0	0	0	0	0	0	0

此 WDT Key 暫存器(WDTKEY,$ 97)爲控制 WDT 控制暫存器 (WDTC,$ 9F)之設定及保護。

當對 WDT Key 暫存器 (WDTKEY,$ 97)先後寫入 1EH 及 E1H 時(須依此順序先後寫入),始可對 WDT 控制暫存器 (WDTC,$ 9F)設定。

當對 WDT Key 暫存器 (WDTKEY,$ 97)先後寫入 E1H 及 1EH 時（須依此順序先後寫入），則 WDT 控制暫存器 (WDTC,$ 9F)已被保護，無法修改。

WDT Key 暫存器- (WDTKEY, $8F)(SM59D03/04G2 設定)

	bit-7							bit-0
	WDT KEY7	WDT KEY6	WDT KEY5	WDT KEY4	WDT KEY3	WDT KEY2	WDT KEY1	WDT KEY0
Read / Write:	W	W	W	W	W	W	W	W
Reset value:	0	0	0	0	0	0	0	0

此 WDT Key 暫存器(WDTKEY,$ 8F)爲控制 WDT 控制暫存器 (WDTC,$ 8E)之設定及保護。
當對 WDT Key 暫存器 (WDTKEY,$ 8F)先後寫入 1EH 及 E1H 時（須依此順序先後寫入），始可對 WDT 控制暫存器 (WDTC,$ 8E)設定。
當對 WDT Key 暫存器 (WDTKEY,$ 8F)先後寫入 E1H 及 1EH 時（須依此順序先後寫入），則 WDT 控制暫存器 (WDTC,$ 8E)已被保護，無法修改。

1.3. 系統控制暫存器 (SCONF, $BF)

	bit-7							bit-0
	WDR	Unused	Unused	Unused	DFEN	ISPE	OME	ALEI
Read / Write:	R/W	-	-	-	R/W	R/W	R/W	R/W
Reset value:	0	*	*	*	0	0	0	0

WDR: Watch Dog Timer 重置位

當系統重置時是借著 WDT 溢位時產生 WDR 將被設定爲 1，初使值 = 0。

2. 程式範例：

2.1. Assembly language：

```
;***********************    SET DATA ***********************************
WDTC        .REG    09FH                ;SFR 內 WDTC 的地址
WDTKEY      .REG    097H                ;SFR 內 WDTKEY 的地址
;WDTC       .REG    08EH                ;SFR 內 WDTC 的地址(59D03/04G2)
;WDTKEY     .REG    08FH                ;SFR 內 WDTKEY 的地址(59D03/04G2)
SCONF       .REG    0BFH                ;SFR 內 SCONF 的地址
;************************************************************************
            ORG     0000H               ;設定目前程式的起始位址
;******************* 判斷 SCONF 的 WDR 是否有 Reset 產生 ***************
reset:
            MOV     a,SCONF             ;將 SCONF 暫存器的內容放至 A 暫存器中
            JNB     acc.7,poweron_reset ;acc.7 的值若爲 0 則跳至 power on_reset
;******************** acc.7 爲 1 時的執行程式(即 WDT Reset 動作) *************
wdt_reset:
            MOV     P1,#00              ;如果進入 WDT RESET 副程式,則 P1 會全亮全滅一次。
            CALL    delay
            ANL     a,#7fh      ;將 SCONF 的 WDR 清除爲 0,爲了做下一次是否被 RESET 的偵測
            MOV     SCONF,a
            MOV     WDTKEY,#1Eh
```

```
        MOV     WDTKEY,#0E1h        ;UnProtect WDTC
        MOV     WDTC,#10000110b     ; 啟動 WDT Timer 且開始計數及選擇內部頻率除頻大小
        MOV     WDTKEY,#0E1h
        MOV     WDTKEY,#1Eh         ;Protect WDTC
        MOV     P1,#0ffh
        CALL    delay
        JMP     start
;******************** acc.7 爲 0 時的執行程式(即設定 WDT 功能) **************
poweron_reset:
        MOV     WDTKEY,#01Eh
        MOV     WDTKEY,#0E1h        ;UnProtect WDTC
        MOV     WDTC,#10000110b     ; 啟動 WDT Timer 且開始計數及選擇內部頻率除頻大小
        MOV     WDTKEY,#0E1h
        MOV     WDTKEY,#1Eh         ;Protect WDTC
;*************** 執行 main program (LED 閃爍程式由 Port 0 輸出) ************
start:
        MOV     a,#55h
        MOV     P0,a
        CALL    delay
        MOV     a,#0aah
        MOV     P0,a
        CALL    delay
        JMP     start
;************************ 執行 Delay 的動作 ****************************
delay:
        MOV     20H,#08h
b3:     MOV     21H,#0ffh
b2:     CALL    clearWDT            ;呼叫 clearWDT 副程式
        MOV     22H,#0ffh
b1:     DJNZ    22H,b1
        DJNZ    21H,b2
        DJNZ    20H,b3
        RET
;****************** 設定 WDTC 的 Clear Bit ******************************
clearWDT:
        MOV     WDTKEY,#1Eh
```

```
MOV     WDTKEY,#0E1h      ;UnProtect WDTC
MOV     a,WDTC            ;將 WDTC 的 CLEAR 設定爲 1,執行清除計數器的動作
ORL     a,#20h
MOV     WDTC,a
MOV     WDTKEY,#0E1h
MOV     WDTKEY,#1Eh       ;Protect WDTC
RET
END
```

2.2. C language：

```
#include "SM59264C.h"
//#include "SM59D04G2.h"
#include<intrins.h>//--use "_nop_();"
void WDT_Enable(void) // Watch Dog enable
{
    WDTKEY   =   0x1E;
    WDTKEY   =   0xE1;
    WDTC     =   0x86;
    WDTKEY   =   0xE1;
    WDTKEY   =   0x1E;
}
void WDT_Clear(void)   // Watch Dog Disable and counter clear
{
    WDTKEY   =   0x1E;
    WDTKEY   =   0xE1;
    WDTC     =   WDTC | 0x20;
    WDTKEY   =   0xE1;
    WDTKEY   =   0x1E;
}
void Delay_x1ms(int time)
{
    int count=0,T_base;
    TMOD=TMOD | 0x01;               // set timer0 mode1
    T_base=0x10000-0x733;   //0x10000-0x733;
    while(count<time)
```

```
    {
      TH0=T_base/0x100;     //catch hi byte
      TL0=T_base%0x100;     //catch low byte
      TR0=1;                //set TR0, start timer1 mode1
      while(TF0==0)         //TMOD bit-7 is TF0 , if TF0=1 overflow
      {
          _nop_();
          WDT_Clear();
       }
      TR0=0;
      TF0=0;                          //clr TF0
      count++;
    }
    TMOD=TMOD & 0xFE;                 // stop timer0 mode1
}
void WDT_Reset(void) // Watch Dog Reset
{
     unsigned char temp;
     temp = SCONF & 0x80;
     if (temp==0x80)
     {
      P1 = 0X00;
      SCONF = SCONF & 0X7F;
      Delay_x1ms(300);
      P1 = 0Xff;
      Delay_x1ms(300);
     }
}
//==============================
void main(void)
{
     WDT_Enable();
     WDT_Reset();
     while(1)
     {
     P0   =    0x55;
```

```
    Delay_x1ms(300);
    P0   =     0xAA;
    Delay_x1ms(300);
    }
}
```